**Lab 1:**

**Introduction to PSOC Creator**

****

**Microcomputer Systems**

**ELC 343 - L1**

**Nikita Eisenhauer and Jake Levine**

**Submission: 9/10/17**

**Introduction**

In this introductory lab, students were tasked with developing an understanding of programmable embedded devices and their uses and functions in modern world. Specifically, students were to become familiar with the PSOC Creator program developed by Cypress for use with their PSOC (Programmable System on a Chip) boards, through the watching of introductory videos. Emphasis was also placed on understanding the reasons why the PSOC Creator environment was developed, the methods by which it interfaces with the PSOC board and how the program enables faster and more efficient development of embedded applications.

**Discussion**

The PSOC architecture integrates several functionalities of devices such as FPGAs and ASICs into a common package providing a middle ground between both extremes. PSOC boards contain a main CPU like processor that interfaces with multiple analog and digital microcontroller units (MCUs) as well as other circuitry built into the board. Each of these units have the potential to be wired in any configuration the user sees fit and thus these small segmented logic blocks can be assembled like puzzle pieces by programming the PSOC board using the PSOC Creator. Signals can be easily and efficiently routed across the board, additionally the particular board students will be using contains a small breadboard upon which external logic chips, transistors and circuitry can be connected. Overall the PSOC architecture allows for quick design implementation and reconfiguration on a larger scale than an FPGA (logic wise) and with a degree of freedom that cannot be done with an ASIC.

The PSOC Creator specifically allows the PSOC board to be interacted with on both a graphical wiring / schematic based approach and text based C language programming. C is used to program the PSOC board CPU to perform time critical interrupts, number crunching and data storage, while schematic design is used for wiring the various analog and digital modules with one another and the CPU. The creator also incorporates code debugging tools to correct sources of error found before and after compilation, as well as a simulator that can provide logic design verification before the PSOC board is even programmed. All of these tools allow an engineer to quickly hash out designs for use with PSOC boards in a short amount of time with minimal overhead costs.

Using the PSOC creator provides insightful information and data regarding a PSOC projects wiring, layout, pin configuration and CPU tasks. Immediately from within the creator the specific PSOC board that the project is intended to work with can be identified from its context menu. Within the tabs menu, the schematic diagram of the system shows exactly what modules are being used by the project and how they are wired to one another, other pins and I/O modules. Details of individual module functions such as the duration of a PWM block or the workings of interrupt logic can be opened in separate windows. Other individual components such as resistors can be further examined to understand their exact operation such as pull-up vs. pull-down resistor. Within the main.c file application specific code which dictates PSOC firmware operation can be seen and edited and other project header files listed within the directory submenus include various other functions that can be used within the main.c program file for operation of the PSOC board.

A real life example of using the PSOC board for analog data processing is in the creation of a simple sampling program, to act as analog to digital encoder. A timer or interrupt module could be programmed to record a voltage reading of an incoming analog waveform to a specific port on the PSOC board. The voltage reading could then be analyzed by the CPU to determine (based off a given threshold voltage number) whether the value correlated to a binary 1 or 0 and then be stored in the PSOC board’s memory for other uses. This is just one example of the various tasks the PSOC architecture could be employed to perform.